УДК 28.23.35

ПРОЕКТИРОВАНИЕ СЛОЖНОЙ СИСТЕМЫ НА ОСНОВЕ ОБЪЕКТНО-ОРИЕНТИРОВАННОГО ПОДХОДА

В.А. Силич, М.П. Силич

Томский политехнический университет
E-mail: vas@osu.cctpu.edu.ru

Предлагается метод проектирования сложных систем, позволяющий объединять различные методики системного анализа и инженерии знаний на базе декларативной модели, основанной на объектной парадигме представления знаний.

Проектирование сложной системы на этапе создания концепции немыслимо без использования аппарата системного анализа. Однако на пути регулярного массового применения системного подхода для решения конкретных практических задач существуют определенные трудности, связанные с отсутствием "технологии" применения системных представлений, принципов, моделей и методов. Эта проблема еще более обострилась в последнее время в связи с усложнением искусственных систем, создаваемых в различных сферах человеческой деятельности.

По некоторым направлениям предпринимаются попытки создания системных технологий проектирования. В частности, в такой передовой области, как проектирование информационных систем уже возникла целая индустрия CASE-технологий, в которых системные представления используются для моделирования программных продуктов с помощью автоматизированных средств поддержки. Большинство из этих технологий используют методологию системного подхода как концептуальную основу построения модели предметной области. Это характерно и для более ранних технологий структурного анализа, ориентированных на методы SADT (Structured Analysis and Design Technique), DFD (Data Flow Diagrams), ERD (Entity-Relationship Diagrams) и др., так и для более поздних объектно-ориентированных технологий анализа/проектирования (Object-Oriented Analysis/Design – OOA/D). Последние переживают период бурного развития, что вызвано в немалой степени созданием объектного унифицированного языка моделирования UML (Unified Modeling Language) и объектно-ориентированных CASE-средств [1, 2].

Несмотря на успехи в развитии CASE-технологий, говорить о создании комплексной системной технологии, пригодной для решения задач проектирования сложных систем различной природы (технических, организационно-технологических, социальных и др.) еще рано. Причинами такого положения дел являются следующие моменты.

1. CASE-технологии в основном предназначены для проектирования информационных систем, в связи с чем они оперируют собственным понятийным аппаратом, ориентированным на данную область. Для обозначения многих понятий системной теории вводятся новые термины, термины же системного анализа используются зачастую некорректно, в новых ракурсах и отношениях. Разработанные языки моделирования вполне приемлемы при построении моделей информационных систем, но не всегда удобны для описания других предметных областей.

2. Системные представления используются в CASE-технологиях в основном для описания свойств автоматизируемой системы, ее отдельных компонент и взаимосвязей между ними. Подобное описание отражает либо существующее состояние системы (модель AS-IS), либо желаемое (модель TO-BE). Нерешенной проблемой при этом остается переход от моделей AS-IS к моделям TO-BE. Другими словами, существующие технологии, как правило, не содержат процедур генерации и выбора вариантов реализации системы и ее отдельных компонент, а также процедур координации решений, принимаемых на разных уровнях представления системы. Большое количество разнообразных прикладных методов системного анализа и теории принятия решений, ориентированных на поиск средств достижения целей системы, оказались невостребованными в рамках существующих технологий.

В то же время, как показала практика, именно в рамках CASE-технологий модели и методы системного анализа приобретают прагматическую силу и широкое распространение благодаря более строгой формализации и поддержке инструментальными автоматизированными средствами. Становление и развитие системной технологии должно идти по пути объединения лучших достижений отрасли CASE-технологий и теории системного анализа.

Особую роль в этой интеграции может сыграть объектно-ориентированный язык моделирования, благодаря предоставляемой им возможности комплексировать декларативное представление знаний с процедурным. Кроме того, объектная методология предоставляет следующие важные преимущества: возможность сборки системы из готовых повторно используемых компонент; возможность накапливать теоретические и опытные знания в виде библиотек классов на основе механизма наследования; простота внесения изменений в проекты за счет использования свойств наследования и полиморфизма.

В данной статье рассматривается один из методов проектирования сложных систем, главной особенностью которого является возможность объединять различные методики системного анализа на базе единой модели, основанной на объектной парадигме представления знаний. При этом для построения модели могут использоваться экспертные знания, описывающие типовые свойства, структуры и закономерности отдельных классов систем, представляемые на языках

инженерии знаний. Предлагаемый метод лежит в основе новой концепции системной технологии [3], позволяющей описывать с единых позиций различные аспекты сложной системы, а также проектировать комплексные системы для разрешения сложных многофакторных проблемных ситуаций.

Основной принцип системного подхода – рассмотрение системы и любой ее компоненты, с одной стороны, как некоторого целостного объекта, описываемого набором свойств, характеристик и выполняемых им действий, а, с другой стороны, как совокупности более мелких компонент (подсистем, элементов), взаимодействующих между собой. Таким образом, мы можем говорить о двух модельных представлениях системы (подсистемы): в виде объектной модели, представляющей собой описание эмерджентных свойств и поведения, и в виде компонентной модели, отражающей совокупность более мелких компонент и отношений между ними. Первое представление соответствует модели "черного ящика", второе – модели структуры.

Выделение компонент осуществляется методом последовательной декомпозиции. В результате сложная система представляется как иерархия подсистем, построенная на основе отношений "часть – целое" ("Part-of"), называемых также отношениями агрегации. По классификации Месаровича [4] это, так называемая стратифицированная иерархия, описывающая систему на различных уровнях абстрагирования, т.е. детальности отражения элементов, свойств, характеристик.

Каждая из подсистем дерева представляется в виде объекта. Как известно, в объектно-ориентированных моделях под объектом понимают информационную структуру в виде совокупности атрибутов (свойств, параметров, характеристик) и совокупности методов (действий, процедур, операций). Каждый объект является представителем некоторого класса однотипных объектов, описывающего общие свойства входящих в него объектов. Описание класса включает: состав атрибутов, для каждого из которых указаны тип данных и область значений, и совокупность методов, общих для всех объектов класса. Конкретный объект, называемый экземпляром, определяется конкретными значениями атрибутов.

Классы могут наследоваться друг от друга, т.е. на основе любого класса можно создавать новые классы по принципу "от общего к частному". Порождаемые новые классы сохраняют все свойства классов-родителей, а также обладают некоторыми новыми собственными свойствами. Наследование позволяет строить иерархии классов, начиная с некоторого простого первоначального предка и кончая более сложными и специфическими объектами. Это иерархии, отражающие родовидовые отношения типа "частное – общее" ("Is-a" или "Kind-of"). Иерархии могут быть объединены в библиотеки типовых объектов. Использование типовых объектов значительно упрощает процесс формирования модели.

Предлагается при формировании модели системы использовать функциональный подход, при котором выделение подсистем осуществляется в соответствии с тем, какую функцию она должна выполнять. При построении стратифицированной иерархии (Part-of-иерархии) общая функция системы, как некоторый процесс, обеспечивающий ее выполнение, декомпозируется на отдельные подпроцессы, выполняющие отдельные функции. Объектное описание подсистемы формируется на основе наиболее подходящего класса из иерархии наследования классов (Kind-of-иерархии), в которой хранятся описания функций разной степени детальности. Например, типовой класс "Функция" содержит такие атрибуты, как "Вход", "Выход", "Инструмент", "Способ", "Длительность", "Стоимость". От него наследуются такие классы, как "Бизнес-функция", "Программный блок" и др., каждый из которых имеет собственный список атрибутов, уточняющий и расширяющий список атрибутов предка.

Объектная модель подсистемы содержит описание элементов, реализующих процесс, происходящий в подсистеме, описание свойств, характеристик этих элементов и описание признаков и свойств всего процесса в целом. При этом модель может содержать различные варианты описания подсистемы, соответствующие различным способам реализации данной подсистемы. Таким образом, модель подсистемы должна содержать класс подсистемы, определяющий структуру ее описания, и множество экземпляров, отражающих различные варианты подсистемы.

Формально модель подсистемы можно описать следующим образом:

$$M^S(S_i) = < C(S_i), \{V_j(S_i)\} >,$$

где $C(S_i)$ – описание класса подсистемы S_i; $\{V_j(S_i)\}$ – множество экземпляров (вариантов) подсистемы S_i. Описание класса содержит множество атрибутов, для каждого из которых задан идентификатор (название), тип и область значений (домен), а также множество методов (присоединенных процедур):

$$C = < \{x_k, t_k, D_k\}, F >,$$

где x_k – идентификатор атрибута, t_k – тип атрибута, D_k – домен атрибута, F – множество методов.

Экземпляры подсистемы формируются на основе классов. Для формирования множества экземпляров и выбора оптимальных могут использоваться различные методы теории принятия решений. Так, для генерации экземпляров можно воспользоваться переборными методами, например, методом морфологического анализа: варьируя возможные значения атрибутов, создается множество вариантов подсистемы, соответствующих различным комбинациям значений. Для оценки и выбора наиболее перспективных вариантов могут быть использованы методы однокритериального и многокритериального выбора (по интегральным критериям различного вида) на основе экспертных оценок.

Еще один способ формирования вариантов подсистем – использование закономерностей, показывающих, как значения одних атрибутов могут быть определены через значения других. Для этого в описание подсистемы необходимо включить каузальные отношения между характеристиками подсистемы, которые можно

трактовать как отражение следующих отношений: “причина – следствие”, “средство – цель”, “условие – заключение”, “аргумент – функция”. Отношения-зависимости устанавливаются между простыми атрибутами, которые будем называть параметрами, и отражают зависимости между значениями различных параметров, описываемые некоторой закономерностью. С помощью закономерностей значения одних параметров определяются через значения других, от которых они зависят. Рассмотрим один из методов формирования и выбора вариантов на основе отношений зависимостей – метод “Функциональные сети параметров” [3].

В соответствии с рассматриваемым методом первым шагом построения модели зависимостей является формирование сети параметров, фиксирующей наличие зависимостей между ними. Из множества параметров $X = \{x_i\}$, описывающих подсистему, выбираются так называемые базовые, значения которых не зависят от значений других параметров. Это первичные, непосредственные характеристики. К их числу относятся “управляемые” параметры, значения которых разработчик может непосредственно задавать, а также “внешние возмущения”, значения которых определяются извне. Остальные параметры непосредственно или опосредованно зависят от базовых. Из небазовых параметров выделяется один или несколько “целевых” параметров, значения которых определяют целевое состояние подсистемы.

Сеть может строиться как от истоков, т.е. от базовых параметров, так и от стоков, в качестве которых выступают “целевые” параметры. Дуги сети отражают факт наличия функциональной зависимости (рис. 1). Каждое отношение может иметь несколько входов и только один выход. В результате получается сеть типа иерархии без циклов и петель. Как правило, сеть имеет вид иерархии, которую по классификации Месаровича [4] можно отнести к иерархии типа слоев.

На содержательном уровне сеть зависимостей параметров можно интерпретировать как:

– сценарий, показывающий, к каким последствиям, выражаемым значениями целевых параметров, приводит ситуация, задаваемая определенной комбинацией значений базовых параметров;
– дерево целей, показывающее, какие подцели (значения промежуточных и базовых параметров) должны быть достигнуты, чтобы достичь цель (заданные значения целевых параметров).

Сеть задает как бы “каркас” модели зависимостей. Затем для каждого небазового параметра x_k задается вид зависимости – закономерность, показывающая, как именно его значения определяются значениями параметров, от которых он непосредственно зависит:

$$x_k = f(x_1, x_2, \ldots x_n),$$

где $x_1, x_2, \ldots x_n$ параметры, от которых непосредственно зависит x_k.

Вид зависимости может задаваться различными способами. Это может быть формула, совокупность правил-продукций, некоторая процедура-функция, настроенная нейросеть и т.д. В объектной модели

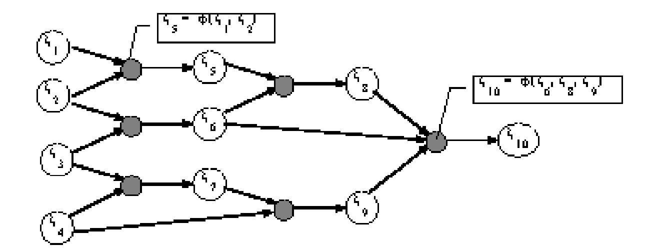


Рис. 1. Сеть функциональных зависимостей параметров

подсистемы отношения-зависимости задаются как объекты, ссылки на которые содержатся в специальных атрибутах. При этом каждое из отношений представляет собой экземпляр класса объекта-зависимости, который включает в себя идентификатор выходного параметра x^{out}, список идентификаторов входящих параметров $\{x_i^{in}\}$, закономерность $f(\{x_i^{in}\})$:

$$g = < x^{out}, \{x_i^{in}\}, f(\{x_i^{in}\}) > .$$

Закономерности, описывающие зависимости между атрибутами подсистемы, могут быть заранее сформированы и сохранены в библиотеке типовых моделей вместе с описанием класса подсистемы.

Для формирования вариантов подсистемы на основе отношений зависимости используют: метод пополнения описания экземпляра, позволяющий на основе заданных значений базовых параметров с помощью закономерностей вывести значения других, а также метод поиска значения заданного параметра. Данные методы применяют различные алгоритмы, в частности, алгоритм прямого альтернативного вывода, алгоритм прямого безальтернативного вывода и др. [3]. Для выбора оптимальных вариантов используется метод оптимизации, позволяющий определять экземпляры подсистемы, задаваемые комбинацией значений базовых параметров, с наилучшим значением критерия и удовлетворяющие заданным ограничениям. Для поиска оптимальных вариантов могут быть применены различные алгоритмы, использующие стратегии как прямого, так и обратного вывода [3].

Перейдем к рассмотрению компонентной модели сложной системы, в которую наряду с объектными моделями подсистем входят различного рода отношения между подсистемами. Будем различать три вида отношений, включаемых в компонентную модель: отношения агрегации (“часть – целое”), отношения ассоциации (взаимосвязи) и отношения зависимости между параметрами различных подсистем. Отношения любого вида так же, как и подсистемы, представляются в виде объектов.

Основу модели составляет дерево подсистем, формируемое путем последовательной декомпозиции исходной системы. Будем называть совокупность подсистем, полученных в результате декомпозиции одной и той же системы предыдущего уровня, подуровнем. При декомпозиции каждая подсистема порождает свой подуровень. Порождающую подсистему будем называть материнской, а подсистемы, составляющие

подуровень – дочерними. Поскольку при переходе от верхних уровней к нижним количество подсистем растет, а значит, растет и число отношений между подсистемами, рассмотрение всех отношений между всеми подсистемами сильно усложняет модель. Целесообразно на каждом уровне рассматривать только отношения между подсистемами одного подуровня и между дочерними подсистемами и материнской системой. При этом отношения между подуровнями учитываются на верхних уровнях, как отношения между подсистемами, порождающими подуровни. Это позволяет понизить сложность модели.

Таким образом, иерархическую модель сложной системы можно предста… совокупность моделей подуровней (sublevel) $M^{sl}(S_i)$, каждая из которых содержит модель материнской системы $M^S(S_i)$, множество моделей дочерних подсистем $\{M^S(S_j)\}$; отношение агрегации R^P, связывающее материнскую систему с дочерними; множество отношений ассоциации между дочерними подсистемами $\{R_k^A\}$ и множество отношений зависимости $\{R_l^D\}$:

$$M^{sl}(S_i)=<M^S(S_i),\{M^S(S_j)\},R^P,\{R_k^A\},\{R_l^D\}>.$$

На рис. 2 представлена модель сложной системы в виде иерархии моделей подуровней. Для простоты показаны только отношения агрегации и отношения ассоциации.

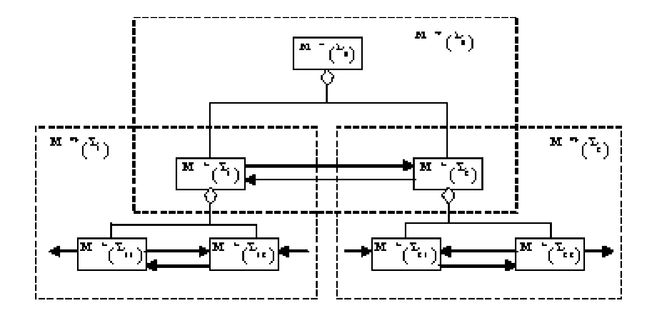


Рис. 2. Модель сложной системы в виде иерархии моделей подуровней

Отношения агрегации, связывающие материнскую систему с дочерними подсистемами, соответствуют такому понятию системного анализа, как “основание декомпозиции”. Существуют стандартные основания декомпозиции. Примеры стандартных оснований декомпозиции для систем организационно-технологического типа (бизнес-систем): “бизнес-система – подсистемы, соответствующие различным видам социальной деятельности (производственная, социальная, экологическая и т.д.)”; “производственная система – подсистемы, производящие различные виды конечного продукта”; “производственная система – подсистемы, соответствующие этапам жизненного цикла производства”.

Отношения ассоциации связывают подсистемы подуровня между собой и с внешними подсистемами. При переходе на следующий, более нижний уровень иерархии внешние отношения могут декомпозироваться, т.е. распадаться на несколько более мелких, или целиком переходить на следующий уровень. К отношениям ассоциации относятся: отношения коммуникации (потоки), заключающиеся в передаче вещества, энергии, информации; пространственные отношения; временные отношения; разнообразные эмпирические отношения. Классы описаний типовых ассоциаций, как и стандартные основания декомпозиции хранятся в библиотеке типовых моделей.

Отношения зависимости устанавливаются между атрибутами подсистем, входящих в подуровень, атрибутами отношений ассоциации, а также атрибутами материнской системы, породившей подуровень. Данный вид отношений используется для согласования вариантов подсистем друг с другом и с вариантом материнской системы. Двухуровневую модель, в которой материнская система, расположенная на верхнем уровне, накладывает ограничения на дочерние подсистемы, расположенные на нижнем уровне, можно отнести к эшелонной иерархии по классификации Месаровича [4]. Модели данного типа называют также координационными.

Будем различать следующие типы зависимостей, включаемых в модель подуровня (рис. 3):

– межуровневые, связывающие параметры подсистем с параметрами материнской системы. Примером является зависимость затрат материнской системы от затрат отдельных подсистем, определяемая как сумма;

– межподсистемные, отражающие взаимное влияние подсистем, входящих в подуровень. Это так называемый баланс входов-выходов подсистем. Пример: объем продукции, производимой некоторой подсистемой, должен быть равен объему продукции, поступающей на вход другой подсистемы, умноженному на коэффициент потерь, возникающих при передаче.

Для нахождения оптимального варианта подуровня, т.е. оптимального сочетания вариантов подсистем, удовлетворяющего требованиям, предъявляемым материнской системой, необходимо выбрать критерий подуровня и наложить ограничения. В качестве критерия эффективности подуровня, как правило, выбирается критерий материнской системы, для которого задана зависимость от локальных критериев подсистем в виде некоторой обобщенной функции:

$$x_{kr}^0 = F(x_{kr}^1,\ x_{kr}^2,\ \ldots),$$

где x_{kr}^0 – критерий подуровня, $x_{kr}^1,\ x_{kr}^2,\ \ldots$ – локальные критерии подсистем.

Ограничения могут быть двух видов: межподсистемные и межуровневые (так называемые, совокупные). В качестве первых выступают межподсистемные зависимости. В качестве вторых – ограничения, накладываемые на параметры материнской системы, которые являются функцией параметров подсистем. Оптимальным является такое сочетание вариантов подсистем, для которого выполняются межподсистемные и совокупные ограничения и при этом критерий подуровня достигает своего наилучшего значения. Для нахождения оптимального варианта подуровня могут использоваться процедуры координации. Авторами разработаны две процедуры координации: метод

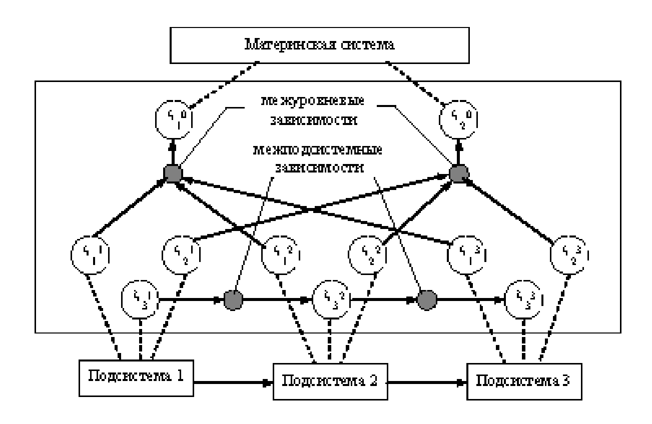


Рис. 3. Модель зависимостей параметров подуровня

"корректировки безусловно оптимального варианта" и "волновой" метод [3].

Для поддержки описанного метода построения иерархической объектной модели сложной системы спроектирован инструментальный комплекс Engine 2.0. Модель сложной системы, формируемая с помощью Engine 2.0, содержит некоторые канонические диаграммы UML, а также некоторые дополнительные диаграммы и методы, позволяющие использовать декларативную модель системы для генерирования различных вариантов реализации системы и ее компонент и выбора наиболее перспективных вариантов. Комплекс позволяет использовать при создании модели библиотеки классов описаний компонент системы, предоставляет возможность одновременной работы над проектом нескольких пользователей, импорт/экспорт проектов и их частей. В настоящее время уже реализована пилотная версия системы.

Изложенная методология предназначена, прежде всего, для проектирования сложных организационно-технологических систем на ранних этапах (на этапах концептуализации). Технология проектирования при этом рассматривается как автоматизированная технология поэтапного формирования комплексной модели сложной системы с использованием экспертных знаний об отдельных видах сложных систем и их компонент, хранимых в виде библиотек типовых моделей. Базовые идеи объектно-ориентированной системной методологии были использованы для решения ряда практических задач, в том числе для совершенствования деятельности Томского технопарка [3, 5], при разработке программы энергетической безопасности региона [3, 6], программы энергосбережения Томской области, программы социально-экономического развития региона [3].

СПИСОК ЛИТЕРАТУРЫ

1. Леоненков А.В. Самоучитель UML. – СПб.: БХВ-Петербург, 2001. – 304 с.
2. Новоженов Ю.В. Объектно-ориентированные технологии разработки сложных программных систем. – М.: Аргус-софт компани, 1996. – 115 с.
3. Силич М.П. Системная технология: объектноориентированный подход. – Томск: Том. гос. ун-т систем управления и радиоэлектроники, 2002. – 224 с.
4. Месарович М., Мако Д., Такахара И. Теория иерархических многоуровневых систем. – М.: Мир, 1973. –344 с.
5. Силич В.А., Силич М.П., Ямпольский С.З. Совершенствование деятельности технопарка на основе системной технологии // Актуальные проблемы управления-2001: Матер. междунар. научно-практ. конф. – Москва, 2001. – С. 70–73.
6. Литвак В.В., Силич В.А., Силич М.П., Яворский М.И. Концепция энергетической безопасности субъектов Федерации // Ресурсы регионов России. – М.: ВНТИЦ. – 2001. – № 2. – С. 32–44.